Research Paper
Int'l J. of Aeronautical & Space Sci. 11(2), 110–117 (2010)
DOI:10.5139/IJASS.2010.11.2.0110

IJASS
International Journal of Aeronautical and Space Sciences

Performance Analysis of Pursuit-Evasion Game-Based Guidance Laws

Young-Sam Kim*

Flight Control Group, MUAV Development Center, Korean Air, Daejeon 305-811, Korea

Tae-Hun Kim and Min-Jea Tahk*****

Department of Aerospace Engineering, Korea Advanced Institute of Science and Technology, Daejeon 305-701, Korea

Abstract

We propose guidance laws based on a pursuit-evasion game. The game solutions are obtained from a pursuit-evasion game solver developed by the authors. We introduce a direct method to solve planar pursuit-evasion games with control variable constraints in which the game solution is sought by iteration of the update and correction steps. The initial value of the game solution is used for guidance of the evader and the pursuer, and then the pursuit-evasion game is solved again at the next time step. In this respect, the proposed guidance laws are similar to the approach of model predictive control. The proposed guidance method is compared to proportional navigation guidance for a pursuit-evasion scenario in which the evader always tries to maximize the capture time. The capture sets of the two guidance methods are demonstrated.

Key words: Pursuit-evasion game, Differential game, Direct method

1. Introduction

The pursuit-evasion game is a kind of differential game demonstrated by Isaacs (1967). It is an important class of two-player zero-sum differential games with perfect information and is useful in military applications, especially in missile guidance applications. In the game, the pursuer and the evader try to minimize and maximize the intercept time or the miss-distance as a payoff function. Among previous studies, Guelman et al. (1988) studied a simple case of pursuit-evasion game that can be solved without addressing a two-point boundary value problem. Breitner et al. (1993) proved that a multiple shooting method is able to precisely solve practical pursuit-evasion games subject to state constraints. In addition to these studies, a variety of parameter optimization methods such as those described in (Hargraves and Paris, 1987) can be extended to solve realistic pursuit-evasion games. In this study, we propose a direct method for solving pursuit-evasion games (Tahk et al., 1998a, b).

We first summarize the algorithm proposed in (Tahk et al., 1998a, b) for the reader's convenience. The algorithm is a direct method based on discretization of control inputs for solving pursuit-evasion games for which the intercept time is the payoff function of the game. Every iteration of the algorithm has two features: update and correction. The update step improves the evader's control to maximize the intercept time, and modifies the pursuer's control to satisfy the terminal condition. After applying the update step several times, the correction step is used to minimize the pursuer's flight time.

Note that the solution of the pursuit-evasion game can be used for guidance of the pursuer and the evader if the solution is obtained in real time. For this purpose, the iteration number of the solver is limited to reduce the computation time, at the cost of solution accuracy (Kim et al., 2006). In this study, the work of Kim et al. (2006) was refined for better numerical efficiency. Also, the capture set of the proposed

*Researcher
**PhD Candidate
***Professor, Corresponding author, E-mail: mjtahk@fdcl.kaist.ac.kr

guidance method is compared with that of proportional navigation to confirm that the proposed differential game guidance laws provide a smaller no-escape envelope.

2. Problem Formulation

The problem is a two-dimensional pursuit-evasion game with the final time as a payoff function. The dynamics of the pursuer and the evader are expressed as follows:

$$\begin{aligned} \dot{x}_p &= f_p(x_p(t), u_p(t), t) \\ \dot{x}_e &= f_e(x_e(t), u_e(t), t) \end{aligned} \tag{1}$$

The admissible control inputs $u_p(t)$ and $u_e(t)$ are assumed to be piecewise-continuous functions subject to the following constraints:

$$\begin{aligned} |u_p(t)| &\le 1 \\ |u_e(t)| &\le 1 \end{aligned} \tag{2}$$

Our pursuit-evasion problem is written as

$$\max_{u_e} \min_{u_p} J = t_f \tag{3}$$

subject to the constraints in Eq. (2) and the final constraint,

$$r_p(t_f) = r_e(t_f) \tag{4}$$

To develop a numerical algorithm, we used a direct approach based on parameterization of the control inputs. The control inputs of both players are separated into the following form:

$$\begin{aligned} u_p &= \left[u_{p,1}\ u_{p,2}\ \cdots\ u_{p,N} \right]^T \\ u_e &= \left[u_{e,1}\ u_{e,2}\ \cdots\ u_{e,N} \right]^T \end{aligned} \tag{5}$$

where $u_{p,k}$ and $u_{e,k}$, which are assumed to be constant in the k-th interval, should satisfy the constraints in Eq. (2) as follows:

$$|u_{p,k}| \le 1,\ |u_{e,k}| \le 1, \quad k = 1, 2, \cdots, N \tag{6}$$

The solution technique proposed in (Tahk et al., 1998b) is composed of two procedures: update and correction. The update procedure determines δu_e, a small perturbation of u_e, in the direction of maximizing t_f^*. It also includes an algorithm for the computation of a new u_p^*. Instead of finding the minimum time solution, this algorithm computes δu_p, the smallest variation of u_p^*, to satisfy the final constraint (Eq. 4). If u_p^* is far from the minimum time solution, the correction procedure executes and provides an adequate tuning of u_p^* to satisfy the optimality within the error bounds defined by the user. The aforementioned procedures, illustrated in Fig. 1, are implemented repeatedly until there is no improvement in t_f^*. In (Kim et al., 2006), the update and correction procedures are called step 1 and step 2 to avoid confusion of terminologies.

2.1 Step 1 (update algorithm)

We assume that a capture within a finite time is guaranteed. Any perturbations of both players' control input, δu_e and δu_p, are said to be admissible if they satisfy following conditions:

$$\begin{aligned} |u_e + \delta u_e| &\le 1 \\ |u_p + \delta u_p| &\le 1 . \end{aligned} \tag{7}$$

The capture condition is expressed as

$$P\delta u_p + v_p dt_f = E\delta u_e + v_e dt_f + \Delta r_f \tag{8}$$

where P and E are defined as

$$\begin{aligned} P &= \frac{\partial r_p(t_f)}{\partial u_p} \in R^{2\times N} \\ E &= \frac{\partial r_e(t_f)}{\partial u_e} \in R^{2\times N} \end{aligned} \tag{9}$$

and v_p and v_e are the sensitivities of $r_p(t_f)$ and $r_e(t_f)$, respectively, to the perturbation of t_f. We define v_r as follows:

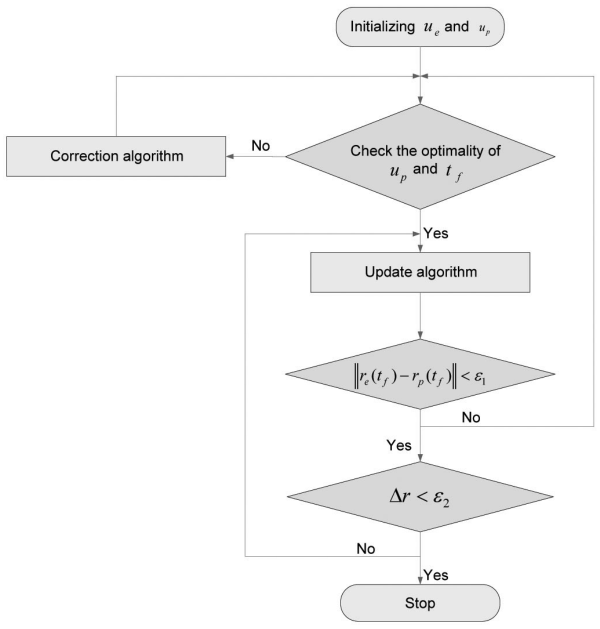
Initializing u_e and u_p
Correction algorithm
No
Check the optimality of u_p and t_f
Yes
Update algorithm
$\|r_e(t_f) - r_p(t_f)\| < \varepsilon_1$
No
Yes
$\Delta r < \varepsilon_2$
No
Yes
Stop

Fig. 1. Flowchart of the algorithm in (Kim et al., 2006).

$$v_r = v_p - v_e \tag{10}$$

Then, Eq. (8) can be written as

$$v_r dt_f = E\delta u_e - P\delta u_p + \Delta r_f \,. \tag{11}$$

Let n_1 and n_2 denote two orthonormal unit vectors in the two-dimensional space. We define n_1 as

$$n_1 = \frac{v_r}{|v_r|} \,. \tag{12}$$

Now, P , E , and $\triangle r_f$ are expressed as

$$\begin{aligned} P &= n_1 p_1^T + n_2 p_2^T \\ E &= n_1 e_1^T + n_2 e_2^T \\ \Delta r_f &= d_1 n_1 + d_2 n_2 \end{aligned} \tag{13}$$

where p_1 , p_2 , e_1 , and e_2 are $N \,{}^{\prime}\, 1$ column vectors and d_1 and d_2 are scalars. For a given δu_e , let δu_p^o , which is orthogonal to the null space of p_2^T , be the minimum norm solution of the combination of Eqs. (11) and (13), which is expressed as

$$\delta u_p^o = p_2 \left(p_2^T p_2\right)^{-1} \left(e_2^T \delta u_e + d_2\right) . \tag{14}$$

If δu_p^o is admissible, then $\delta u_p^* = \delta u_p^o$. For $u_p = u_p^*$ and $t_f = t_f^*$, δu_p satisfying the capture condition can be written as

$$\delta u_p = \delta u_p^o + \delta u_p^n \tag{15}$$

where δu_p^n is the vector in the null space of p_2^T, which can be chosen freely without affecting the capture condition. After some manipulations of Eqs. (14) and (15), we obtain

$$|v_r| dt_f = g_e^T \delta u_e - p_1^T \delta u_p^n - p_1^T p_2 \left(p_2^T p_2\right)^{-1} d_2 \tag{16}$$

where

$$g_e^T = e_1^T - p_1^T p_2 \left(p_2^T p_2\right)^{-1} e_2^T . \tag{17}$$

The evader's control input is then updated as follows:

$$u_{e,k}(j+1) = sat\left[u_{e,k}(j) + \alpha g_{e,k}(j)\right] \tag{18}$$

where j denotes the iteration number, and subscript k is kth element of the separated evader's control variables.

When the deviation of u_p and t_f from the minimum norm solution grows too much, an optimal δu_p^n should be found to correct u_p and t_f. δu_p^n, an undefined term of Eq. (16), is calculated as follows. We write δu_p^n as

$$\delta u_p^n = -\delta u_p^v + \delta u_p^r \tag{19}$$

where δu_p^v is the amount of constraint violation of $u_p + \delta u_p^o + \delta u_p^n$, and δu_p^r is used to make δu_p^n which belongs to the null space of p_2^T . Given δu_p^v, δu_p^r can be expressed as

$$\delta u_p^r = \delta u_p^{ro} + \delta u_p^{rn} \tag{20}$$

where δu_p^{ro} belongs to the null space of $\hat{p}_2$, and δu_p^{rn} is to be determined in an optimal way.

For minimizing $dJ \equiv -p_1^T\left(-\delta u_p^v + \delta u_p^{ro}\right)$, we find the δu_p^{ro} that minimizes the norm of $-\delta u_p^v + \delta u_p^{ro}$. The norm of $\delta u_p^n = -\delta u_p^v + \delta u_p^{ro}$ is minimized by

$$\delta u_p^{ro} = \hat{p}_2 \left(\hat{p}_2^T \hat{p}_2\right)^{-1} p_2 \delta u_p^v \tag{21}$$

After calculating δu_p^n, each component of $u_p + \delta u_p^o + \delta u_p^n$ is examined to check whether or not any violations of control input constraints occur. As soon as δu_p^n is determined, u_p is updated using the following relationship:

$$u_{p,k}(i+1) = u_{p,k}(i) + \delta u_{p,k}^o(i) + \delta u_{p,k}^n(i) \tag{22}$$

Once u_e and u_p are updated, t_f is updated using

$$|v_r| dt_f(i+1) = d_1 + e_1^T\left[u_e(i+1) - u_e(i)\right] - p_1^T\left[u_p(i+1) - u_p(i)\right] \tag{23}$$

In Eqs. 22) and (23), *i* denotes the iteration number.

2.2 Step 2 (correction algorithm)

In step 1, the pursuer's control input is determined to satisfy the capture condition, and $u_p + \delta u_p$ is assumed to close to the optimal control variable for the case of small perturbations of both players' control variables. If the pursuer's control input u_p is far from the optimum to minimize a payoff function, we optimize the pursuer's control variable during the correction procedure.

Assume that the pursuer's control input is subject to variation, but the evader's control input is fixed; that is, $\delta u_e = 0$. The first variation necessary condition for optimality is given by $dJ_p = -p_1^T \delta u_p = 0$ for all admissible δu_p . To inspect the optimality condition at a later time, we define I_S as the set of saturated parameters. It is represented as

$$I_s = \left\{k \mid \left|u_{p,k}^*\right| = 1\right\} \tag{24}$$

where K_S and $s(i)$ denote the number of elements in I_S, and its ith element. We construct a $N \times K_s$ matrix S as follows:

$$S_{ij} = \begin{cases} -1 & \text{if } j = s(i) \text{ and } u^*_{p,j} = +1 \\ +1 & \text{if } j = s(i) \text{ and } u^*_{p,j} = -1 \\ 0 & \text{if } j \neq s(i) \end{cases} \tag{25}$$

The pursuer's control variable is restricted to constraints, and a saturated control variable can be perturbed only in one direction, as shown by

$$S^T \delta u_p \geq 0 \tag{26}$$

Then, let q_1 denotes the component of p_1 normal to p_2, which is given by

$$q_1 = p_1 - \frac{p_2^T p_1}{p_2^T p_2} p_2 \cdot. \tag{27}$$

Hence, the Kuhn-Tucker condition is written as

$$q_1 + \lambda^* p_2 + \mu^{*T} S = 0 \tag{28}$$

where $\mu_k > 0$ for all $k = 1, \cdots, K_s$. We also derive the following necessary conditions for optimality:

$$\begin{aligned} q_{1,k} &= 0, & k &\notin I_s \\ q_{1,k} u^*_{p,i} &\geq 0, & k &\in I_s \cdot \end{aligned} \tag{29}$$

For the correction procedure, we set $\delta u_e = 0$ as previously mentioned. Then, we substitute all elements of q_1 that do not satisfy the necessary conditions for optimality by zero to obtain $\hat{q}_1$, and we compute $\hat{q}_1$, the component of $\hat{q}_1$ normal to $\hat{p}_2$:

$$\tilde{q}_1 = \hat{q}_1 - \frac{\hat{p}_2^T \hat{q}_1}{\hat{p}_2^T \hat{p}_2} \hat{p}_2 \tag{30}$$

We choose δu_p^{rn} as

$$\delta u_p^{rn} = \beta \hat{q}_1, \quad \beta > 0 \tag{31}$$

to reduce t_f because $\hat{q}_1$ is the null space of p_2^T. We compute δu_p^{rn} and the total correction of δu_p given by the following relations, starting from $\delta u_p^o = \delta u_p^v = \delta u_p^{ro}$, as follows:

$$\begin{aligned} \delta u_p &= \delta u_p^o + \delta u_p^n \\ \delta u_p^n &= -\delta u_p^v + \delta u_p^{ro} + \delta u_p^{rn} \end{aligned} \tag{32}$$

If there is any violation of the control input constraint, we use the update procedure of u_p as explained in step 1.

3. Constructing the Proposed Guidance Law

In this section, we explain the method we used to construct the guidance law proposed in (Kim et al., 2006). The algorithm introduced in the preceding section consists of two steps. This algorithm initializes both players' control inputs, and then checks the optimality conditions. If the pursuer's control input and final time do not satisfy the optimality conditions, step 2 is performed; otherwise, step 1 is executed. The evader tries to maximize the final time or to avoid capture by the pursuer, and the pursuer tries to intercept the evader. After performing step 1, we examine whether a difference in the positions of the players at the final time is within the error bound. If it is within the error bound, a variation of the evader's position at the final time is examined. If the variation is within the error bound, the algorithm is terminated and gives optimal trajectories. Otherwise, step 1 is repeated. If the difference in the positions of the players at the final time is not within the error bound, the optimality conditions of the pursuer's control input and the final tim e are checked again.

The initializations of both players' control inputs are the same as the algorithm proposed in (Tahk et al., 1998a, b). In this case, the algorithm is started with step 2. At the proposed algorithm, step 2 is executed when a violation of the optimality condition has occurred and step 1 is repeated when a variation of the evader's position at the final time is not within the error bound. To construct a guidance law, this procedure is changed to execute step 2 once and step 1 ten times. The dynamics of both players are integrated with specific integration time interval Δt using the initial value of the game solution after performing step 2. Then, state variables changed by integration are considered as initial conditions. The termination condition is the same as the proposed algorithm. The algorithm as a guidance law is described in Fig. 2.

With integration, this modified algorithm changes control inputs. First, control input elements of both players are got rid of the control input sets after an integration and the others are replaced previous elements of each control inputs. The reason is that the first control input elements of the pursuer and the evader are used during integration, and both players' positions are updated by integration. By reason of integration with Δt, both players have new initial conditions. After executing integration and replacement (movement) of control inputs, the control input elements of both players are improved during step 1 as previously mentioned. In this respect, the proposed guidance law is similar to the approach of model predictive control (MPC). A schematic diagram of the movement of control input elements is shown in Fig. 3. In addition, we reduced the integration time interval Δt when t_f / N was smaller than the specific integration time interval Δt. In this case, time interval Δt was reduced to t_f / N. If Δt is used consistently, pursuit-evasion game solutions are calculated with the small time interval compared to Δt. In this case, integration with Δt can result in a large miss distance because the dynamics of both players integrate with a much longer time interval than with a discrete time interval of their control inputs near the final time. Thus, we should reduce the integration time interval, which guarantees that integration time interval Δt is close to t_f / N. If the integration time interval continues to decrease, Δt becomes infinitesimal and the iteration executes infinitely. To prevent this, the integration time interval is fixed the specific value near the final time and the iteration number of the solver is limited. Applying this technique, we can improve the interception precision at the final stage. This guidance law is called the differential game guidance law.

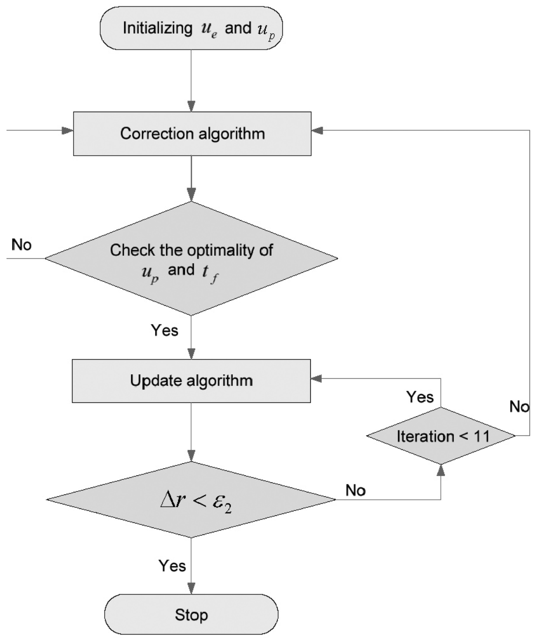
Initializing u_e and u_p
Correction algorithm
No
Check the optimality of u_p and t_f
Yes
Update algorithm
Yes
No
Iteration < 11
$\Delta r < \varepsilon_2$
No
Yes
Stop

Fig. 2. Flow chart of the algorithm after modification (Kim et al., 2006).

4. Numerical Simulation

The differential game guidance law proposed in (Kim et al., 2006) wasapplied to the planar pursuit-evasion game problem illustrated in Fig. 4. The equations of motion of the evader are described as

$$\begin{aligned} &\dot{x}_e = v_e \cos\gamma_e \quad , \quad \dot{y}_e = v_e \sin\gamma_e \\ &\dot{\gamma}_e = \frac{v_e}{R_e} u_e, \qquad |u_e| \le 1 \\ &\dot{v}_e = -\frac{v_e^2}{R_e} c u_e^2 \end{aligned} \tag{33}$$

In these equations, x_e and y_e denote the position in Cartesian coordinates, γ_e is the flight path angle, v_e is the speed, u_e is the non-dimensional control input variable, R_e is the minimum turn radius, and c is the drag coefficient. If the drag coefficient is zero, the evader does not produce a drag force and its speed does not decrease.

The equations of motion of the pursuer, which are analogous to those of the evader, are as follows:

$$\begin{aligned} &\dot{x}_p = v_p \cos\gamma_p \quad , \quad \dot{y}_p = v_p \sin\gamma_p \\ &\dot{\gamma}_p = \frac{v_p}{R_p} u_p, \qquad |u_p| \le 1 \\ &\dot{v}_p = -\frac{v_p^2}{R_p} \left(a + b u_p^2\right) \end{aligned} \tag{34}$$

where R_p is the minimum turn radius, and x_p, y_p, γ_p, v_p, and u_p are analogous to the evader model. The c-3ptnstant a, which is composed of the zero-lift-drag coefficient C_{D_0} and maximum lift coefficient $C_{L_{\max}}$, is defined as

$$a = \frac{C_{D_0}}{C_{L_{\max}}} . \tag{35}$$

The constant b consists of the maximum lift coefficient and the induced drag factor K:

$$b = K C_{L_{\max}} \tag{36}$$

The engagement scenario for the numerical example is shown in Fig. 5. The evader is initially 6,724.97 m apart from

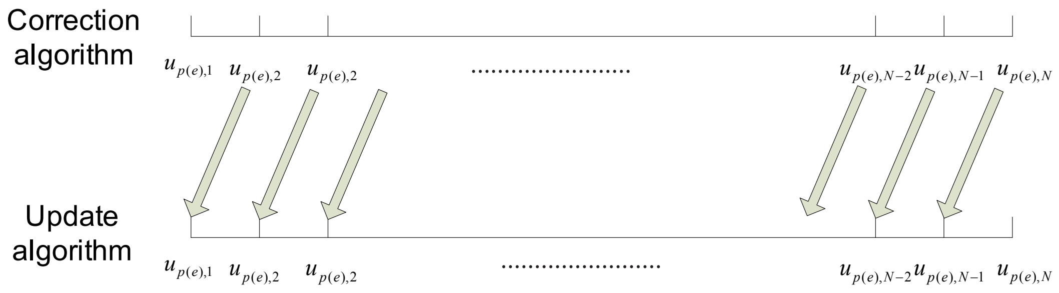
Correction algorithm
$u_{p(e),1}$ $u_{p(e),2}$ $u_{p(e),2}$ .... $u_{p(e),N-2}$ $u_{p(e),N-1}$ $u_{p(e),N}$
Update algorithm
$u_{p(e),1}$ $u_{p(e),2}$ $u_{p(e),2}$ .... $u_{p(e),N-2}$ $u_{p(e),N-1}$ $u_{p(e),N}$

Fig. 3. Movement of the control input elements (Kim et al., 2006).

the pursuer along the x-axis, and moves with $v_e(0)=300m/s$ and $\gamma_e(0)=131.86^\circ$. The pursuer initially moves with $v_p(0)=920.83m/s$ and $\gamma_p(0)=85.36^\circ$. The parameters used in this engagement model are taken from (Guelman et al., 1988) as $R_e=600m$, $R_p=1515.15m$, $a=0.0875$, $b=0.4$. For a realistic evader, the drag coefficient was chosen as 0.4.

The differential game guidance law parameters were chosen as N = 50 and ε_1 = 0.002m. The algorithm proposed in (Tahk et al., 1998a, b) includes the parameter ε_1, but the differential game guidance law does not define it. The reason is that the proposed guidance law; i.e., the differential game guidance law, does not require it because of the difference in the execution control between step 1 and step 2, as mentioned in the previous section.

The trajectories of the pursuer and the evader are shown in Fig. 6. The pursuer intercepts the evader at 18.565769 seconds. The time histories of the control input of both players are illustrated in Fig. 7. The figure shows that an initial control input of the evader is a zero. The figure also shows that the control inputs of both players are constrained by Eq. (2).

To compare the performance of the differential game guidance law with other guidance laws, proportional navigation guidance (PNG) was adopted as a guidance law of the pursuer. In spite of using PNG, we assumed that the evader knows the maneuvers of the pursuer which are optimal. In this case, the pursuer's control inputs are calculated from

$$a_{cmd}=N'V_c\dot{\theta} \tag{37}$$

where N' denotes the effective navigation ratio, and V_c and $\dot{\theta}$ are the closing velocity and line-of-sight rate, respectively. From a_{cmd}, the control input of the pursuer is determined as follows:

$$\dot{\gamma}_p=\frac{v_p}{R_p}u_p=\frac{a_{cmd}}{v_p}\quad,\quad u_p=\frac{R_p}{v_p^2}a_{cmd} \tag{38}$$

It is also constrained by the first expression in Eq. (2). Using PNG as a guidance law, the termination condition is replaced by the miss distance. If the miss distance is smaller than 1 m, we believe that the pursuer intercepts the evader.

Trajectories of the pursuer using PNG as a guidance law,

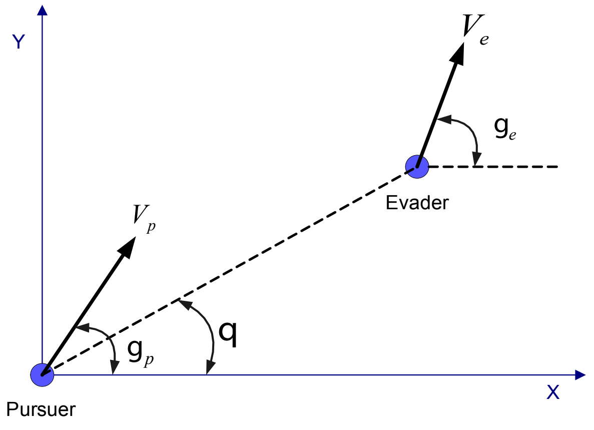

Fig. 4. Pursuer and evader engagement geometry (Kim et al., 2006).

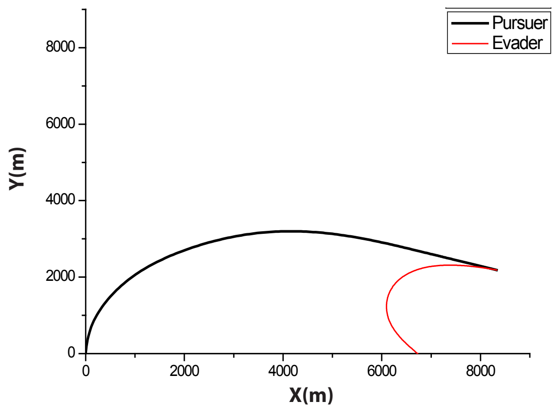

Fig. 6. Trajectories of the pursuer and evader.

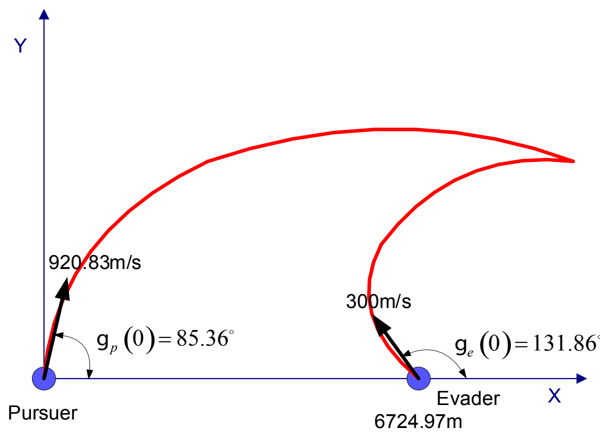

Fig. 5. Intercept scenario (Kim et al., 2006).

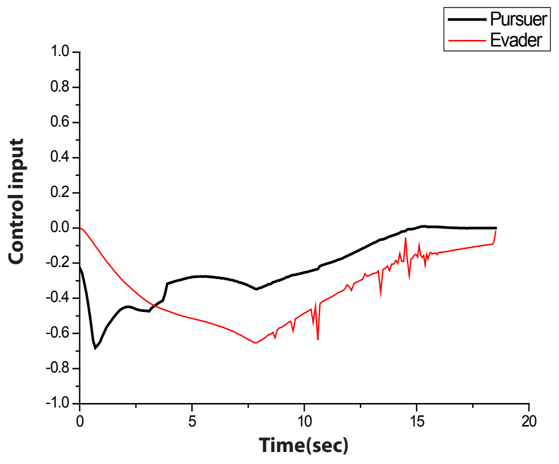

Fig. 7. Time histories of the control input of both players.

and of the evader, are given in Fig. 8. The pursuer captured the evader at 23.339497 seconds. Time histories of both control inputs are shown in Fig. 9, and are similar to Fig. 7. The pursuer's initial control input is a few different between the differential game guidance and PNG. This is because a control input of the pursuer using the proposed guidance law is initialized by PNG, and then it is improved by step 2.

Table 1 is a summary of the simulation results. The intercept time of the differential game guidance law is shorter than the time using PNG. In the case of using PNG, a computation time is approximately three times longer than that using pursuit-evasion game solution. The number of iterations needed to capture the evader is nearly two times greater than that of the proposed guidance law.

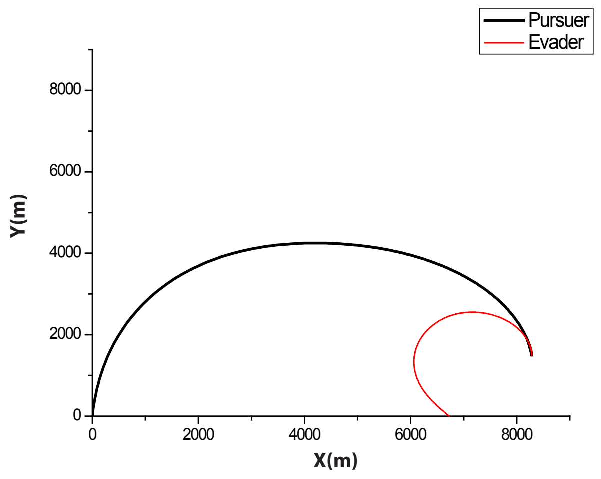

Fig. 8. Trajectories of the pursuer using PNG and the evader.

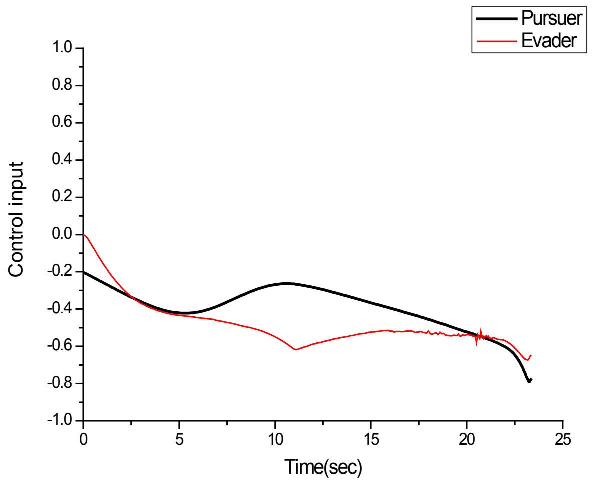

Fig. 9. Time histories of the control input of the pursuer using PNG and the evader.

Table 1. Summary of simulation results

Pursuer's guidance law	Iteration	Computation time	Intercept time
Based on pursuit-evasion game solutions	1,937	8 m 34 s	18.565769 s
Proportional navigation	3,797	23 m 27 s	23.339497 s

To compare the interception performance of both methods, which are both players using the differential game guidance law and the pursuer using PNG, the capture set was calculated. For simplicity, we considered only the initial condition for the previous example except that the initial flight path angles of the pursuer and the evader varied from 0° to 20° and from 0° to 350°, respectively. The reason why we selected the initial flight path angle ranges of the pursuer is that seeker's gimbals used typical guided missiles adopt lock-on type. The conventional field of view of a seeker is limited from 2° to 5°, but the field of view of seeker-adopted lock-on type gimbals is larger than 15°. The field of view of gimbaled seekers, however, is not much larger than 15°. Thus, the initial flight path angle ranges of the pursuer were chosen as 0° to 20°.

Figures 10 and 11 show the capture sets of the proposed guidance law, and the case using PNG as a guidance law of the pursuer. In these figures, the initial conditions marked with ● are those for which the pursuer can capture the evader within a finite time. The other symbols indicate the termination of the program without a capture. In Fig. 10, the symbol ★ indicates a circumstance for which the pursuer cannot capture the evader under the termination condition of the differential game guidance law, but is able to intercept the evader under the termination condition when using PNG as a guidance law of the pursuer. The symbol × implies that the pursuer is not able to intercept the evader under both termination conditions. In Fig. 11, the symbol × indicates that the miss distance is larger than 1 meter, so interception by the pursuer does not occur. From Figs. 10 and 11, we know that the capture set of the proposed guidance law is larger than one using PNG as a guidance law of the pursuer.

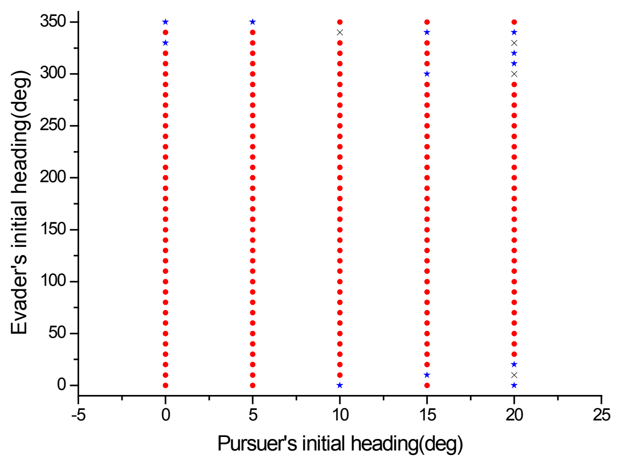

Fig. 10. The capture set of the proposed guidance law.

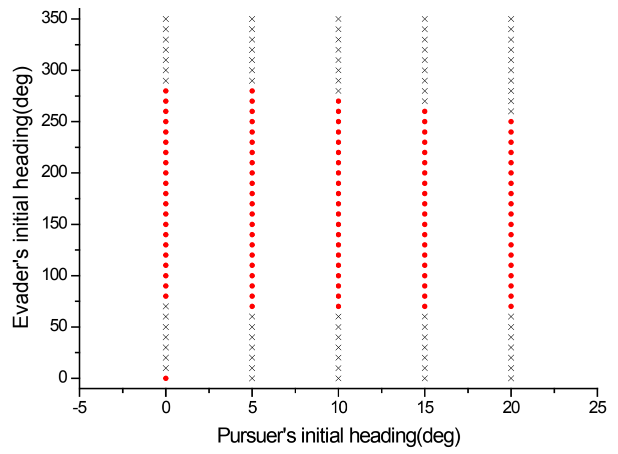

Fig. 11. The capture set of the case using proportional navigation guidance.

5. Conclusions

We carried out a performance analysis of the proposed guidance law based on pursuit-evasion game solutions that were sought by the pursuit-evasion game solver. The derivation of the proposed pursuit-evasion game solver was described, and construction of the proposed guidance law was explained in detail. The guidance law proposed in (Kim et al., 2006) consists of two procedures, update and correction, called step 1 and step 2. Control inputs of the pursuer and the evader were

initialized in the differential game guidance law, and then improved by step 1 and 2. Step 2 executes first that this procedure provides a way to optimize the pursuer trajectory when it deviates excessively from the optimal trajectory. Then, the dynamics of both players were integrated with a specific time interval. Step 1 performs that control inputs of the evader were updated to maximize the increment in the capture time, while the pursuer tried to minimize it. This procedure iterated until satisfying the termination condition. The differential game guidance law was applied to solve a numerical example. For a comparison of the interception performance, PNG was adopted to a guidance law of the pursuer. Simulation results of the engagement scenario were tabulated for each case--the differential game guidance law, and using PNG as the guidance law of the pursuer. Capture sets of both cases were calculated for performance analysis. Based on our numerical example simulation results, we know that the differential game guidance law provided a smaller no-escape envelop than PNG.

References

Breitner, M. H., Pesch, H. J., and Grimm, W. (1993). Complex differential games of pursuit-evasion type with state constraints, part 1: Necessary conditions for optimal open-loop strategies. *Journal of Optimization Theory and Applications*, 78, 419-441.

Guelman, M., Shinar, J., and Green, A. (1988). Qualitative study of a planar pursuit-evasion game in atmosphere. *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Minneapolis, MI. AIAA Paper 88-4158.

Hargraves, C. R. and Paris, S. W. (1987). Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics*, l0, 338-342.

Isaacs, R. (1967). *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*. 2 printing ed. New York, NY: Wiley.

Kim, Y. S., Tahk, M. J., and Ryu, H. (2006). A guidance law based on pursuit-evasion game solutions. *KSAS-JSASS Joint International Symposium on Aerospace Engineering*, Busan, Korea.

Tahk, M. J., Ryu, H., and Kim, J. G. (1998a). An iterative numerical method for class of quantitative pursuit-evasion games. *Proceeding of AIAA Guidance, Navigation, and Control Conference*, Boston, MA. pp. 175-182.

Tahk, M. J., Ryu, H., Kim, J. G., and Rhee, I. S. (1998b). A gradient-based direct method for complex pursuit-evasion games. *Proceedings of the 8th International Symposium on Dynamic Games*, Vaals, Netherlands. pp. 579-582.